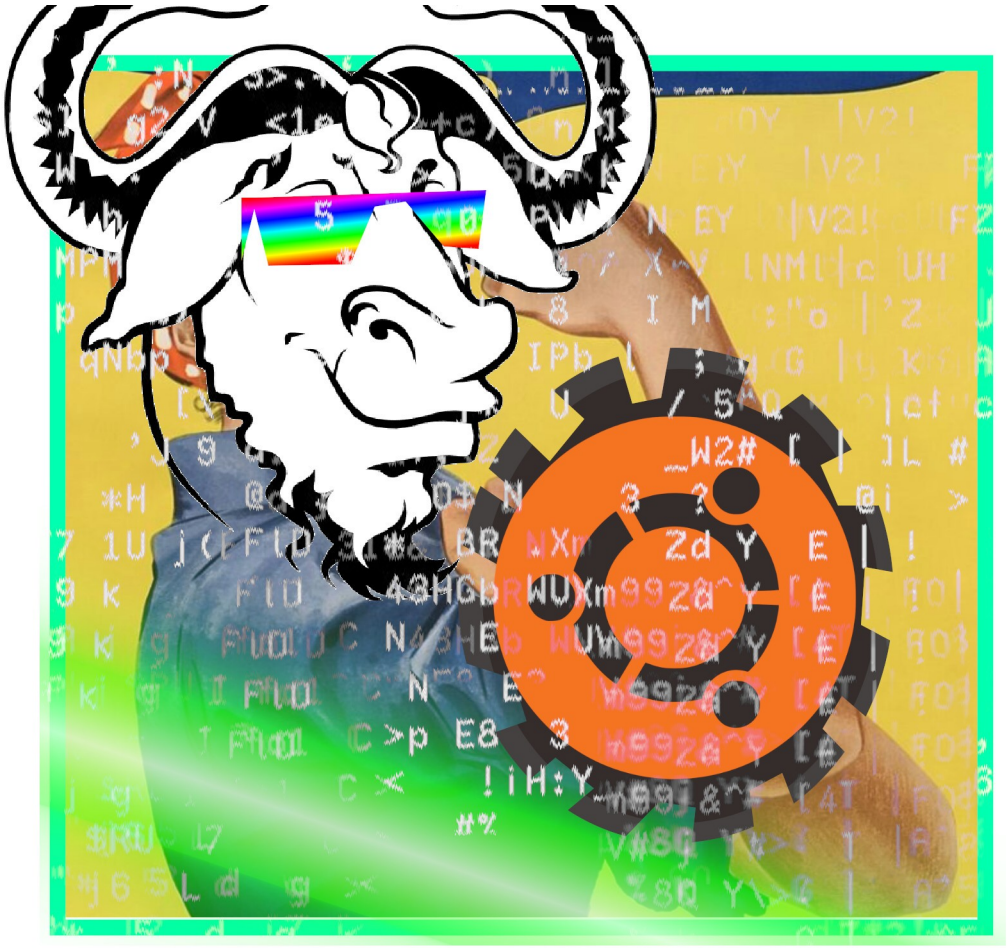
>The Unofficial Ubuntu Manual..for the man



An introduction to **Ubuntu OS** with some of the most important and useful things about it. **Ubuntu is a free operating system** based on the **Linux kernel** and **many open source programs** usualy **GNU/GPL** programs and other **free licenses**. Anyone is free to use / copy / redistribute **Ubuntu**. There are different flavors of the **Ubuntu OS**, like **Xubuntu** or **Lubuntu** that are redesigned with another user interface and/or are better suited for older computers (they need less memory etc), but they are still **Ubuntu** (same applications, or setup with few changes).
This manual is suited for the newcomer noob and anyone else.

'**root**' is the **admin of the system (super user)**, the system has groups (system groups or users) and can have multiple users with their own separate sessions/folders and files.

_
{

Written by Chob and Artemis (шооьака ат зепѝратьаи дот орг)
Thanks to Michell Mickolini, Gen’sEnvy, Bassedul,
Stewie and Donger, wo0GieWoO, Q91, Spud, peach,
oxfuxxx, dnsk, Minnen, idleman, joew771, crax23,
Zorg, Roflcoptr, Linuxyo, BigDadE, Raccoon(BG),
Dumbguy, Helldragon, LOD, Rafunman, Genesis, Gum,
Kaito, TwisTN, Lamule, Muse, Fanny, DragonRift,Wumpus,
Wgreenhouse, DJPH, Xboner, Havenwood, Apeiros,
ArtifexTerra, fant0men, SlammerHedge…
The free internet platforms,
The Open Source Software devs and maintainners,
Анд мѝ шѝни метал ьут.

}

1) How to create a Ubuntu USB boot stick (and boot from it)

Download the live **Ubuntu ISO** from: https://ubuntu.com/#download

Choose the Desktop version.
If your computer is older or if you need a lighter version:

https://xubuntu.org/download/ or https://lubuntu.me/downloads/

(an ISO is an image of a disk that can be written to a USB stick or a DVD etc)

Note about booting a USB stick: when the computer start you may have to press F12 or F2 in order to get the boot menu selection (to choose USB) or eventually you need to go in the BIOS to set the USB boot, before the hard drive/SSD.

Windows users:

Download a tool to write the ISO to the USB stick,
for example: **Rufus** https://rufus.ie/en/

You can then plug your USB stick and start rufus;
You select your ISO file in the rufus app, and then you can click the start button.
The configuration options should be detected automaticly, but if your USB stick doesn't boot later, it means you'll have to tweak the option like: partition scheme / MBR and the target: BIOS or UEFI.

Android users: (tablet or phone)

If your phone / tablet is not rooted you most likely cannot write the ISO from it, however things may have changed, so look in the **google playstore** or **f-droid** or **apkpure store/site** for an app that can write an ISO to USB.
Your phone/tablet must support **OTG**, and you need a micro USB(type C) to female USB adapter.
(**Termux** app may be able to write a USB boot sitck on rooted phones)

Mac users:

You need the **Etcher** app (or eventually another one)
convert the ISO to DMG (that is mac format) with this command:
hdiutil convert "/path/to/ubuntu.iso" -format UDRW -o "/path/to/ubuntu.img"
(it will append .dmg extension automaticly)
Then in the Etcher app, Select Image > and choose the newly created .dmg file.

Linux users:

Run the startup disk creator (sytem tools > startup disk creator)
command: **usb-creator-kde**
or **usb-creator-gtk**
then select the ubuntu ISO file (click Other / Select)

then click '**Make Startup Disk**'

-

Just for information, to do a direct transfer of the ISO file
to a device on linux, can be done with the '**dd**' command:
dd if="/path/to/ubuntu.iso" of="/dev/device_path"
with more options (see **man dd**), however it would not work
for a USB stick because of USB boot mode and partitionning.

2) Quick view of the file system, folder and files

Ubuntu uses ext4 filesystem and may use one or more partition, it can read most other filesystems like Windows FAT/VFAT or NTFS or Android SD cards filesystems etc.

The Ubuntu Linux files on the hard drive or SSD:

"/" (slash) ← is the base of all files and folders (known as 'root' like the admin name)

"/home/username/" ← is the folder of the user (named "username" here)
all the user data, configuration, session, downloads etc are located here,
so if you need to backup your things that's the folder to save,
every user has his own folder into "/home/",
In your home folder are hidden files starting with a dot '.',
like '.config/' or files like '.gtkrc', they are configuration
files for your session, you can list all files including hidden
ones with the command '**ls -lah**'

Then all the other folders are system related:

"/etc/" ← main configuration and boot scripts for the system
"/usr/bin" ← main binaries (executable files) that users can use
"/usr/sbin" ← main binaries for the root (admin) of the system
"/usr/lib" ← the library objects that the executable needs
etc…

Here is a quick view of them:

bin -> usr/bin	: system executable binary programs, any normal program installed with apt should be around here
boot	: boot config and kernel, like 'grub' bootloader things
dev	: all the devices and special files, ex: "/dev/sda" is the hard drive or SDD a USB gamepad could be "/dev/js0", etc
etc	: all system configuration and boot time scripts
home	: every users of the system has a folder inside "/home"
lib -> usr/lib	: libraries that executables needs
lib32 -> usr/lib32	: libraries that executables needs
lib64 -> usr/lib64	: libraries that executables needs
libx32 -> usr/libx32	: libraries that executables needs
lost+found	: file system internal report (if hard drive has an issue or whatever)
media	: USB Sticks or CDs or Phone storage can be accessible here, Ubuntu will 'mount' then in that location, ex: "/media/username/USB-STICK-1/"
mnt	: another 'mount point' that can be used to access DVD or network storage
opt	: this one may hold programs or data compiled by a user (with admin access) or eventually other optionnal things
proc	: contains system infos and reports, about CPU or temperature or technical things (special files)
root	: the home of the admin (root)
run	: system special folder (running entries, services ...)
sbin -> usr/sbin	: executable binaries of root (admin only)
snap	: snaps are self contained packages, some programs are packaged

using this methods, and they are here instead of usual /usr/bin etc

srv : server related folder

swapfile : swapfile is a special file that can be used like RAM (memory for the programs to run), it can be 4 GB or more

sys : special system folder, a little bit like "/proc", it contains system informations

tmp : temporary folder, everything in this folder will disappear at each boot, system and programs are using it to store temporary information/files

usr : this one contains many things, an interesting one is "/usr/share/doc" that has documentation of installed programs, there is like said before executable in "/usr/bin/" or libs "/usr/lib/" you can find some dictionnaries in other folders or wallpapers, fonts etc

var : temporary information and logs of the system, most important is "/var/log" where you can find kern.log and dmesg for system information, for example if you plug a USB device, then you look into dmesg, you should see some information about your device

By default Ubuntu will automount a USB stick or phone storage etc to make it accessible to you, however to manually access a filesystem for one reason or another (for a network storage for example or an ISO file) you need the **'mount'** command(**man mount**), you need to be **root** to use it usualy, here is few examples:

(all the commands here must be run as **root** or with "**sudo**" before them)

Mount the first partition of the 2nd drive to "/mnt" folder:

mount "/dev/sdb1" "/mnt"

note: sda is first disc, sdb is second disc etc, and the number after it, is the partition number:
"/dev/sda1" ← is the first partition(1) of the first disc(sda)

Unmount it:

umount "/dev/sdb1"
or
umount "/mnt"

Mount a .iso file:

mount -o loop -t iso9660 "/path/to/file.iso" "/mnt"

Mount and unmount a .cue .bin file: (needs to install fuse)

fuseiso -p image_file.bin "/path/to/mount"
fusermount -u "/path/to/mount"

Mount and list a NFS: (nfs-common and nfs-kernel-server services must be installed and running...)

exportfs 192.168.43.27:/mnt
mount -t nfs 192.168.43.27:mnt /mnt
showmount -e 192.168.43.27

Mount a SMB(cifs or smb): (needs to install cifs-utils)

mount -t cifs -o username=smbuser //192.168.1.100/documents /mnt

or, as read only (use the link directly mount.smb3 instead mount)

mount.smb3 -o username=anakita,ro //192.168.43.27/mnt ./boom

3) Basic shell/console commands

Ubuntu has a text mode you can go to text mode usualy with **control + alt + F1** or **F2** … FX and back to graphic mode with **control + alt + F1** or F7. It's called a **terminal** or **console mode** and you can enter commands, run scripts etc. In graphic mode you can run **xterm** or **gnome-terminal** or **qterminal**, **rxvt** to enter commands (CLI) (some of them needs to be installed though).
The program that will read your command that runs in textmode or inside xterm, qterm etc is known as the 'shell' (usually it is '/bin/sh', '/bin/bash' or '/bin/dash'), it is this program that reads the commands you type and display the prompt, text output or run the scripts (shell scripts – list of commands).

And here is a list of few basic and usefull commands:

(for every command or program you can usualy do: **command –help** or **command -h** to have help about it, and to run a program or script that is located in your current folder (non system script or app) add "**./**" in front of it like: "./command" or "sh" like "sh ./command")

FILESYSTEM AND BASICS:

echo : display some text (ex: echo "hello world")
cat : display the content of a file (ex: cat "/etc/os-release" ← display OS release file)
ls : list folders and files of current directory (ex: ls /tmp ← will list the /tmp folder)
cd : change current directory (ex: cd /tmp ← go to the tmp directory)
(cd .. <- go to parent directory)
(cd ~/ <- go to your home directory)
pwd : displays current working directory (normaly your "/home/username" when you first start the shell
cp : copy a file (ex: cp "file.txt" "file-backup.txt" ← make a backup copy)
mv : move a file(ex: mv "file.txt" "./BACKUP/file-backup.txt" ← move the file to the BACKUP folder)
rm : delete(remove) a file (ex: rm "./file.txt")
mkdir : create a directory (ex: mkdir "Backup")
chmod : change file mode (see next chapter)
chown : change file owner (see next chapter)
useradd : add a new user (see next chapter)
userdel : delete a user
groupadd : create a new group
groupdel : delete a group
sudo : execute a command as root (ex: sudo dmesg)
su : switch user (ex: su username2)
ps : displays the list of running programs (known as process)
(ex: ps fax ← process tree list)
top : realtime view of the running processes and CPU/memory usage in table form
note: the RAM usage reported might not be meaningful at first,
you should refer to other tools also if you don't know (see "free" and others)
free : displays current memory status (RAM and swap)
df : displays disks/storage usage (disk free), (ex: df -h ← human readable disks usage space)
du : disk usage, displays the size of a folder (ex: du -sh "/home/username" ← size of the home directory of username)
head : displays the header lines of a file
tail : displays the bottom lines of a file

vi : a text editor
nano : a text editor that is more friendly than vi
grep : a tool to extract or filter text string from a file or command line.
A command line output can be manipulated by one or more other commands using the '|' character (pipe), very usefull in text mode, especially with grep
ex: cat /etc/os-release | grep ubuntu
It will only displays lines of that files that contains the term 'ubuntu'
sed : replace/manipulate strings, (usefull with pipe also, like grep)
ex: echo "hello world" | sed "s/hello/ubuntu/"
It will print "ubuntu world" because sed will replace the world "hello" by "ubuntu"
find : search paths (ex: find "/etc" ← list all folders/files entries in "/etc")
more : stop the output at every screen (usefull with pipe too), if a command outputs so many lines that it does not fit the screen you can use 'more' and it will pause at every screen until you press space, ex: "find /etc | more"

Stream redirect: when you run a command like "**ls -l**" you can redirect the output to a file automaticly instead of printing it onscreen, ex:

ls -l > /tmp/filelist ← this will redirect the screen output (named STDOUT) to the file /tmp/filelist

However, error messages also printed onscreen are using 'STDERR' and would not be redirected so if you want to also redirect them to the file you will add '**2>&1**' after the first file redirection:

ls -l > /tmp/filelist 2>&1 ← redirect all screen outputs (STDOUT and STDERR) to the file

(2 = ERR and 1 = OUT, it redirect error to out then to the file)
(names: standard input / output / error : STDIN / OUT / ERR)

SYSTEM:

Note: you may need root access for the next commands, you can prepend "**sudo**" ex: "**sudo dmesg**" (**sudo** will ask you the root password to run the command if needed)

dmesg : displays system log outputs
lspci : displays hardware informations about the computer (graphic/chipset/network/...)
lsusb : displays USB informations/configuration and connected devices
fdisk : display/mamipulate hardrive partitions (ex: fdisk -l /dev/sda ← list partition of disk)
fsck : scan/fix/repair a broken disk partition (ex: fsck /dev/sda1)

PACKAGES:

apt-get : main command to install or remove programs (ex: apt-get install gimp ← install gimp drawing/photo editor)
apt-cache : search for programs (ex: apt-cache search gimp)

Few usefull apt commands: (see packages install chapter later)

apt-get install PROGRAM_NAME ← install a program
apt-get remove –purge PROGRAM_NAME ← remove a program and all it's data
apt-get update ← update the list of available packages

dpkg : install/check a .deb package manually instead of going via apt
(ex: dpkg –get-selections ← list installed packages on the system)
(ex: dpkg -i “mypackage.deb” ← install a .deb package manually)

SERVICES:

service : manage background services of the system (ex: service bluetooth restart <- to restart the bluetooth daemon) (ex: service –status-all ← list status of all services)
systemctl : this is the second command to manage running background services,
(ex: systemctl list-unit-files ← list services)
(ex: systemctl start sshd ← start the ssh server)

NETWORK:

netstat : displays network information (if servers are running, open/listening ports, connections…)
(ex: netstat -anpo ← list all, with numbers and process IDs ...)
ip : informations/configuration about network interfaces/address (ex: ip address)
ifconfig : same as ip (ifconfig is older), it needs to be installed (net-tools, ...)
iwconfig : same as ip/ifconfig but for wireless (WIFI)
nmcli : control the network manager
nc : netcat, network swiss knife tool, can connect or listen to an address (telnet like)
iptables : configure and displays the firewall
ufw : configure and displays the firewall (needs to be installed with apt)
wget : download a file from the web (ex: wget https://address)
curl : same as wget

Many other commands are not listed here, check the folders in **$PATH** (echo **$PATH**), and other online references for Ubuntu CLI (command line interface).

4) Users, groups, file access/owners

The administrator is known as '**root**', it is possible to execute a command with admin rights using the '**sudo**' command (ex: sudo dmesg). The '**su**' command can be used to switch to another account (ex: su anakita). There are groups also, (see file "/etc/groups"), some of them are system related and others can be used to manage a group of normal users (folder access or whatnot).

sudo : execute a command with admin root privileges
su : switch to another user

A file or a folder has read/write and/or execution permissions, for the owner of the file or the group, and others, ex: (username is "abitosh" here, output of "ls -l")

drwxrwxr-x 2 abitosh users 4096 Dec 4 16:20 texts ← this one is a folder
-rw-rw-r-- 1 abitosh users 6 Dec 21 04:07 text.txt ← this one is a text file
-rwxrwxr-x 1 abitosh users 6 Dec 21 04:07 script.sh ← this one is a script

(from left to right)
d = weither the listed entry is a directory (d)
rwx = the read, write, execution permissions for the user (owner)
rwx = the read, write, execution permissions for the group
rwx = the read, write, execution permissions for others
0 = an ID
abitosh = the username (owner)
users = the group
000 = the size of the file (4096)
date/time = the date and time of last modification (Dec 21 04:07)
filename = the file or folder name (script.sh, …)

To change the owner and group of "text.txt" to user "raccoon" and group "pirates",
i would use: (chown == change owner)

chown raccoon:pirates text.txt

(chown user:group filename)

(You can use the command "groupadd" to create a new group, ex: groupadd videogames)
(You can use the command "groupdel" to delete a group)

Then to modify the permissions of the file, we need to tell if it is for the user(**u**) the group(**g**) or the others(**o**), and then the read (**r**), write(**w**) and execution(**x**):
(note that execution is only interesting for scripts and binary executables)

-rw-rw-r-- 1 abitosh users 6 Dec 21 04:07 text.txt ← this one is a text file

chmod u-rwx text.txt ← remove read/write/execution for the user (though it had no exec sets)

----rw-r-- 1 abitosh abitosh 6 Dec 21 04:07 text.txt

chmod g-w text.txt ← remove write for the group

----r--r-- 1 abitosh abitosh 6 Dec 21 04:07 text.txt

chmod o+rwx text.txt ← set read/write/execution for anyone (you don't do that usualy of course)

----r--rwx 1 abitosh abitosh 6 Dec 21 04:07 text.txt

Instead of using this syntax it is possible to use the numeric version of the **chmod** command to change the permissions of the files (ex: chmod 677 text.txt), so you can often find the numeric version syntax of chmod inside scripts or onto online documentation/tutorials, check the manpage for more info: **man chmod**

5) Configuring network

Tools used for network configuration / lookup / scan:

ifconfig : displays or configure network interfaces
(you need to install it, because it's not pre-installed anymore, it's been replaced by 'ip', to install it: apt-get install net-tools)
ip : like ifconfig (default one)
iwconfig : same as ifconfig but for wireless interfaces (WIFI, related tools: iwscan, iwlist, ...)
ping : basic tool to tell is an IP/host is up on the network
netstat : displays network connections, port status etc
nmcli : control the network manager
nc (netcat) : a tool to listen or connect on TCP or UDP (like telnet)
iptables : linux firewall command configuration
ufw : another configuration for iptables / netfilter with more accessible commands,
(this one needs to be installed: apt-get install ufw)
(if you want the graphic interface: apt-get install gufw)
openvpn : tunneling app for virtual private network
(also needs to be installed: apt-get install openvpn)
nmap : a network scanner to check open ports and running services
(same for nmap: apt-get install nmap)

(Ethernet)----------

When plugging an ethernet cable it should automaticly get the IP using DHCP,
then you can check your network interfaces with:

ifconfig -a

or

ip address

You will see the interfaces names like eth0 or wlan0121 and your IP like 192.168.X.X or 10.0.X.X the netmask 255.255.X.X etc.

You can put an interface up or down using:

ifconfig eth0 up or **ifconfig eth0 down**

For pointopoint ethernet connection you can use on both machines:
(pointopoint: connecting one computer to another computer directly with just 1 cable (cross cabled - as opposed to straight cabled when using an ethernet HUB))

ifconfig eth0 192.168.0.1 netmask 255.255.255.0 pointopoint 192.168.0.2 up
(and you reverse the IPs on the second machine of course for the command)

(you may have to specify the broadcast address in some situation by adding something like **broadcast 255.255.255.255** before "pointopoint ...")
(note: "netmask" is the sub-network area, "broadcast" helps spreading the message/packets to all concerned machines and "pointopoint" tells the direct destination link, see network related references for more infomations)

(Ping)----------

then you can try to ping the machine with the ping command:
(the ping command will tell if you got a response from the machine on the other side)

ping 192.168.0.2

if your connected on the internet, use a test host:

ping wikipedia.org

(DNS servers)----------

the DNS servers are used to resolv name<>address, normally it's autoconfigured using your network configuration, the file "/etc/resolv.conf" is created automaticly and contains the IP/s of the DNS servers, however you can force to something else (but it will only be temporary):

echo nameserver 8.8.8.8 > "/etc/resolv.conf" # this will add a line to that file with the google servers here 8.8.8.8 you can have a look at **openDNS** servers or others.

Note that some censoring are done by blocking the DNS from resolving the name<>IP sometimes.

(Netstat)----------

To display information about network activity on your computer, see what port is listening or what connection is going on you can use netstat:

netstat -anpo

this one will display all connection(a) using number notation(n) with process(p) etc, check "man netstat" for more info

(WIFI)----------

Using the network manager client (**nmcli**),

See the list of saved connections:

nmcli c

List available hotspots:

nmcli d wifi list

Connect on a known / saved network: (SSID is the network hotspot name)

nmcli d wifi connect *SSID*

Connect on a new network:(replace SSID, PASS, WLAN_INTERFACE for your net)

nmcli d wifi connect *SSID* password *PASS* iface *WLAN_INTERFACE*

You can also check the underlying tools like iwlist / iwconfig:

scan for hotspots:

iwlist *WLAN_INTERFACE* scan

Connect with something like:

iwconfig *WLAN_INTERFACE* essid *SSID* keys:*MYPASSWORD*
(check **man iwconfig** for more info)

(VPN)----------

To connect to a VPN using a config file and OpenVPN, you can use:

openvpn --connect-timeout 9 --server-poll-timeout 9 --config /path/to/configfile
(with some timeout here)

and if you need to disable IPv6: (which will force IPv4 use)
(sometimes the VPN can be up but the system/app would use the IPv6 or things like that...)

sysctl -w net.ipv6.conf.all.disable_ipv6=1
sysctl -w net.ipv6.conf.default.disable_ipv6=1
sysctl -w net.ipv6.conf.lo.disable_ipv6=1

(Firewall)----------

The firewall on Linux is controlled with **iptables** and there is another app named **ufw** and the graphic version **gufw**, (ufw and gufw needs to be installed with apt)

Here is few usefull **ufw** firewalling commands:

Reset the firewall:

ufw --force reset

Disable default in/out - block all connections:

ufw default deny outgoing

All incomming blocked:

ufw default deny incoming

Allow the **tun0** interface to communicate (usefull with VPN):
(**tun0** is the interface that VPN usualy create for **tun**neling the connection)
(if everything was blocked like previous example)

ufw allow out on tun0 from any to any
ufw allow in on tun0 from any to any

Allow an IP from any interface to outside:

ufw allow out from any to 136.0.0.90

Enabe ufw:

ufw enable

Status:

ufw status

6) Configuring and using a network printer

Weither it is a wired or wireless network printer the first thing is for it to get connected / to connect to it. You can connect the printer to your local network (using a HUB, or to your WIFI hotspot) or (for a wireless printer) put the printer in open hotspot mode (default mode usually), in that case you look into the available WIFI networks on your computer and connect to it. -refer to the manual of the printer for that part-

Now you can type the command '**system-config-printer**' that will open the printer configuration manager of Ubuntu, in that tool you will also be able to see the printer jobs etc.

Now, click '**Add**'

You will be able to choose different modes of installation,
If you know the IP of the printer (It can be displayed/configured on the printer screen config or you can find the default one on the manual etc) it is a good idea to use:

'**Add by URL**',

then in the form, enter the printer IP, like for example "**192.168.223.1**"

Note: if you are connected to the printer in direct mode, without internet access, you need to disconnect the printer and to connect to the internet instead, Ubuntu will not find your printer of course, but it will be able to download things needed to install it, so that when you connect back to your printer later it will be fine.

Then select the model of your printer in the list, or a compatible one.

Then reconnect to your printer if needed and try to print a test page (from **system-config-printer**)

Here are usefull tools and informations about printing:

Print a .jpg from terminal by converting it to postscript:

```
convert file.jpg file.ps
lp file.ps
```

Select the default printer:

```
lpoptions -d printer_name
```

Print the current terminal text:

use the shortcuts: CTRL + PRINT_SCREEN (on rxvt unicode for example (urxvt))

List all printers:

```
lpstat -p -d
```

If something goes wrong with your printer or if you forget the WIFI password, you can factory reset it, look in the manual or on the internet to find the manipulation to do it (like keeping 'cancel' and 'wifi' button pressed for few seconds etc)

7) Write a script (boot time script)

Scripts can be useful to automate tasks at boot time or to rename lots of files etc, so here is a short introduction about scripting on Ubuntu.
Scripts runs in the shell/terminal they can be written in shellscript (sh/bash) in Ruby (Ruby language) or Python (Python language), or others.
Note that shellscript is the traditionnal scripting language on Ubuntu, however it's a bit old and does not offers all the possibilities of Python or Ruby. You should install them using:

apt-get install ruby python

Now. here is a script example that ask the user for a base filename to make multiple copy of a file with an identifier: (same script written in shellscript, then python then ruby)

Enter this command first, before trying the script:

echo hello > mytestfile.txt
(to create the test file used by the script)

SHELL SCRIPT (SH/BASH):

```
#!/bin/sh
# the previous line indicate the interpreter to use (sh), it MUST be the first line
# all the lines with a starting '#' are comments
# variable are defined like: myvar="hello"
# and needs to be preceded with '$' sign when used, like: echo "$myvar"
echo "Please enter base filename:"
# read the input from keyboard into variable 'bfname'
read bfname
# test if the original file exists, note: the '!' means 'NOT' see: "man ["
if [ ! -f mytestfile.txt ]; then
	echo "Sorry, mytestfile.txt does not exist, please create it."
	# quit
	exit 1
fi
# make copies of the file in a loop with a number, variable is 'i'
for i in 0 1 2 3 4 5; do
	echo "Copying to $bfname$i.txt"
	cp mytestfile.txt "$bfname$i.txt"
done
echo "Done"
exit 0
```

PYTHON SCRIPT:

```
#!/usr/bin/env python
# the previous line indicate the interpreter to use (python), it MUST be the first line
# the next line is needed to have access to OS functions
import os
# read the input from keyboard into variable 'bfname'
bfname = input("Please enter base filename:")
# test if the original file exists
```

```
if(os.path.exists("mytestfile.txt") == False):
	print("Sorry, mytestfile.txt does not exist, please create it.")
	# quit
	exit(1)
# make copies of the file with a number, variable is 'i', .. 0 to 5 == 6
for i in range(6):
	nn = bfname + str(i) + ".txt"
	print("Copying to " + nn)
	os.system("cp mytestfile.txt " + nn)
print("Done")
exit(0)
```

RUBY SCRIPT:

```
#!/usr/bin/env ruby
# the previous line indicate the interpreter to use (ruby), it MUST be the first line
puts "Please enter base filename:"
# read the input from keyboard into variable 'bfname'
# (chomp removes the new line (return key) from bfname)
bfname = STDIN.gets.chomp
# test if the original file exist
if(File.exists?("mytestfile.txt") == false) then
	puts "Sorry, mytestfile.txt does not exist, please create it."
	# quit
	exit(1)
end
# make copies of the file with a number, variable is 'i'
0.upto(5){ |i|
	puts "Copying to #{bfname}#{i}.txt"
	system("cp mytestfile.txt #{bfname}#{i}.txt")
}
puts "Done"
exit(0)
```

Now, to run one of the previous scripts, said you named the script file '**myscript.sh**' (or **myscript.py** for python or **myscript.rb** for ruby), you can do:

chmod +x myscript.sh
(to make it executable)

then

./myscript.sh
(to actually start it)

(or you can also do: **sh ./myscript.sh**)
(or for python: **python ./myscript.py**)
(or ruby: **ruby ./myscript.rb**)

And you will have an output like this:

Please enter base filename:
testxyz
Copying to testxyz0.txt
Copying to testxyz1.txt
Copying to testxyz2.txt
Copying to testxyz3.txt
Copying to testxyz4.txt
Copying to testxyz5.txt
Done

And you can check that the files have been created correctly: (for example)

ls -l testxyz0.txt
-rw-rw-r-- 1 abitosh abitosh 6 Dec 21 18:30 testxyz0.txt

Now if you plan to make some scripts it is a good idea to create a "**/home/username/bin**" or "**/home/username/scripts/**" directory (replacing username with your own of course) to place them inside that folder, then you can add that folder into the execution path of Ubuntu to make them accessible from any location, the system variable to do that is **$PATH**, it has the directory entries like "**/bin**" or "**/usr/bin**" where the Ubuntu programs are located, so you can add your script folder to it. In order to do that, edit the file **/home/username/.bashrc** and at the end of the file add the line "**PATH=/home/username/bin:$PATH**" then save the file, and logout/login back, now all the scripts in that folder will be available any time, there is autocompletition with the tab key, and they are resolved first (says if you have a 'ls' script in /home/username/bin it will be used instead of /bin/ls)

Note that the "**/home/username**" is abbreviated "**~/**" and the associated shell variable is **$HOME**
For instance, **echo Hello > /home/username/world.txt** or **echo Hello > ~/world.txt** means exactly the same, in both case it creates the file world.txt in the home folder, so **"~/"** is used everywhere in scripts or tutorials usually.

"/" is known as root (like the admin name/login) and is the base of the filesystem.

The **"."** means current directory, ex: **ls -l ./.bashrc** will list the file '**.bashrc**' (hidden file because it starts with a '.') in the . directory (hence the 2 dots in ./.bashrc)

The **".."** means previous(parent) directory, the parent directory of "/home/username" is "/home" but you may not have the permission to list its content.

The **"*"** in bash can represent any file or folder, it's a joker, ex: "ls *.sh" will list all the .sh files (note that in the case of a folder it could list the inside of the folder like: ls /v* will list the content of /var)

If you want to put your script at session start when login in, you can put it at the end of the ~/.bashrc file. But Ubuntu also has a graphic UI session start manager where you can add your scripts, look in the preferences in the session settings.

A system boot time script could be added in /etc/rc.local (as root) with execution mode (+x) and enabling the rc-local service using: "sudo systemctl enable rc-local.service"
(start it manually with "sudo systemctl start rc-local.service" or "sudo service rc.local start)
However this one may not work anymore and system level script may needs to use the rcX folders entries (like real services HTTP servers etc)

8) Install and manage packages (programs / apps)

You can install or remove a program using the default graphic installer, like **Synaptic**.
and the main command line tool to do it is **apt**. You need to be **root** to install or remove a program, (you can use **sudo** in front of the command line to do it, ex: **sudo apt-get install featherpad**)

Search for a calculator program:

apt-cache search calculator

Install firefox:

apt-get install firefox

Remove firefox and clean up:

apt-get remove --purge firefox

Update:

apt-get update

A lower level package manager(used by apt) is '**dpkg**' that deals with the '**.deb**' files:

Install a .deb package:

dpkg -i chromium-nightbuild.deb

List current packages states: (all packages installed with apt or dpkg)

dpkg --get-selections

Some packages are now distributed using either 'snaps', 'flatpacks' or 'appimage',
to start an appimage file, you usualy just need to make it executable (chmod +x file.appimage) then to start it (./file.appimage).
The snaps or flatpacks should be managed transparently, but we won't talk about it right now, anyway, there is not much of them for now, anyway, you can list installed flatpacks using:

flatpak list -d --app --runtime

Some other programs or scripts can be made available on '**git**' only or other source distribution, in that case, for **git** here you would need to have it first (sudo apt-get install git), then clone the repository of the script/program: **git clone 'https://the.git.program'** and then run it with python or others or try to see how it needs to be compiled/built, however building programs can be quite complicated (scripts don't need compilation).

WARNING: INSTALLING PROGRAMS FROM UKNOWN SOURCES OR UNVERIFIED PROGRAMS CAN BE DANGEROUS FOR YOUR SYSTEM AND FILES.
(That is if you launch an installation from an unknown archive with root privileges, it can destroy your system or crypt your files or install a backdoor)
- however the Linux OSes are known to be of the most safer on this side of things -

9) Use tar, 7zip, unrar

Tar:

Archives on linux are often .tar and it is an historical format, you can find tar.gz (compressed with gzip) or tar.bz / tar.bz2 (compressed with bzip), you only need to use the command line tool 'tar' to create or decompress them. It can also be supported directly by some file manager in graphic mode.

Creating an archive file in tar.gz format from the Documents/ folder:

tar zcvf Doc-archives.tar.gz Documents
(z = gzip, c = create, v = verbose infos, f = archive file)

Then if you want to list the content of the newly created archive:

tar ztvf Doc-archives.tar.gz
(t = list files)

And to extract the archive back:

tar zxvf Doc-archives.tar.gz
(x = extract files)

If you want to use bzip/bzip2 instead of gzip, use 'j' instead of 'z'
Then have a look at 'man tar' if you want more details about it.

7zip:

7z is a tool that can handle lots of zipped file formats, including **.zip** of course, it needs to be installed using:

sudo apt-get intall p7zip p7zip-full

You can create a zip archive file using:

7z a Doc-archives.zip Documents
(a = add)
(This will create the .zip file from the Documents/)

and if you want to extract it:

7z x Doc-archives.zip

And **7z -h** for more infos or **man 7z**

Unrar:

.rar is also a popular format and there is **unrar-free** that can extract them:

unrar -x archive.rar

10) Dosbox/Wine/VirtualBox to emulate older OS or Android

Ubuntu can run and emulate old computers and devices, using different programs, old consoles and arcade games can be emulated with **Retroarch**:

sudo apt-get install retroarch
sudo apt-get install libretro-*

Then you'll have to look around the UI that is similar to a recent console interface.

There are other emulators like Z**snes** for the Super Nintendo and so on, that are older.

Also, if you are looking for a dedicated gaming experience, there are USB sticks that can boot up with a linux system customized for gaming only: (not Ubuntu)

-Recalbox
-Batocera

They includes network support, wireless gamepad connection etc, but we won't talk about them here.

Back on Ubuntu, another gaming gear is **qjoypad**, you need to plug your gamepad then run **qjoypad** and it will put an icon in the tray, double click the icon then you can configure the gamepad to act like keyboards keys, for example for **Freedoom**, if you want to play it with a gamepad or for other game that takes only keyboard as input, can be usefull for **Wine** (windows) games or **DOSbox** games etc.

DOSBox can run DOS programs and games, so just install it with **apt** and check the .bat startup and configuration to run your favorite oldies.

Wine (Winehq) is a compatibilty layer interface that is able to run Windows programs on Ubuntu, it can run many of the old or more recent Windows programs and games, there is an official website with the complete listing of compatible games and programs and also tips and tricks to run others:

https://appdb.winehq.org/

Finaly VirtualBox is a program that can emulate a PC completly, and you can install almost any operating system on it, like if it was a completly different computer (hard disk RAM etc):
(for example if you wanna try other linux distribution safely or to run an old Windows or Android etc)

https://www.virtualbox.org/

11) Most wanted apps and games

SYSTEM:

QPS	: Graphic version of PS, view process, RAM CPU usage...
Blueman-manager	: Bluetooth manager
Gparted	: Partition manager to modify/format/resize hard drive / SSD...
Baobab	: Shows disk usage in graphic form, to find out large folders or files easily
Thunar file manager	: File manager explorer
Brasero	: Burn (write) CDs and DVDs
Python	: Scripting language
Ruby	: Scripting language
OpenVPN	: VPN tunneling anonymous/security (bypass censoring)

TEXT / READER / OFFICE:

Featherpad	: Basic text editor
LibreOffice	: Complete office suite with writer, calc, draw
Abiword	: Text writer
Kiwix	: Read zim archive files (like offline wikipedia)
Qcomicbook	: Comic book / manga / CBR / PDF reader

GRAPHIC:

The GIMP	: 2D picture/photo editor with layers, filters, and many tools
Inkscape	: Vector graphic app
Krita	: 2D painting program with brushes, comics/cartoon/manga oriented
Blender3D	: Full featured 3D modeler/animation/renderer
Image Magick	: Command line program to convert and modify image files

AUDIO:

Pulse Audio Volume Control	: Volume and audio manager
Audacity	: Complete audio editor (edit wave or mp3 files, transform, cuts, ...)
Ardour	: Music creation program
Hydrogen	: Music creation program (drumbox, patterns, samples)

PLAYER / MULTIMEDIA / STREAMING / VIDEO:

OpenShot	: Video editor, fading, resize, cut, join, filtering
VLC	: Multimedia player audio, video, streams, DVD, blurays (avi, mp4, mkv, mp3, flac)
SMPlayer	: Another Multimedia player, audio, video, streams, DVD, blurays (avi, mp4, mkv, mp3, flac)
Ffmpeg	: Command line program to transcode video or audio files

INTERNET:

Firefox	: Web browser
Chromium	: Web browser
Epiphany	: Web browser
Falcon	: Web Browser
Hexchat	: IRC chat client (LIBERA etc)
Mirage / Quaternion / Spectral	: Matrix chat clients
Qbittorrent	: Torrent client to download and share peer to peer

P2P archives and files on the internet

Transmission : Another torrent client
Thunderbird : Mail client
Gftp : FTP client

GAMES:

Godot engine : Game engine for video game developers
Qjoypad : Bind keyboards keys to a joypad/gamepad
Armagetron Advanced : 3D tron cycle game with online mode
Frozen Bubble : Shot bubbles (puzzle bobble)
Gnome-Mahjongg : Mahjong game
Retroarch : Multi system emulator that can run old consoles like Atari, Nintendo, Sega, Sony, ...
GNGB : Gameboy emulator
Nestopia : NES emulator
Zsnes : Super Nintendo emulator
Freedoom / prboom : Open source version of the Doom engine FPS, can run original Doom games and comes with 2 free episodes (phase 1 and 2)
OpenArena : 3D FPS game based on the open source version of Quake 3 Arena engine with new content and online play
Nexuiz : Original 3D FPS game with online play

-See Wine and DOSBox for Windows and DOS games
-There are other emulators for old computer games (CPC/C64/Amiga...)

OTHER:

VirtualBox : Runs a virtual machine PC, DOS, WINDOWS, LINUX, BSD, MAC, ANDROID...
Wine : Runs Windows 9x, vista, 7 programs and games
Dosbox : Runs DOS program and games
Arduino IDE : Eletronic enthousiasts I.C. programming
TOR : Darknet and censor bypass

12) Issue : unsupported hardware or bug? forum and chat community

You can check on the official site **https://ubuntu.com/** and register, in order to report/track bugs etc, and also check the forum for answers or ask questions on **https://ubuntu.com/community** and **https://ubuntuforums.org/**

On top of that there is the chat community on **LIBERA** (**irc.libera.chat**) you can connect to it using **Hexchat** app for example or via a webclient like **mibbit.com** in your web browser, then join the **#ubuntu** channel (by typing "/join #ubuntu") or the derivatives like #xubuntu or #lubuntu etc. There is also other chat rooms like #ubuntu-discuss or #ubuntu-offtopic, and many other open source software related rooms, for example if you have an issue with Blender3D you can join the #blender room

Then there are also **Discord** chat rooms, and **Matrix** chat rooms.

Another good resource site for troubleshooting thing is **stackoverflow**, though it is more oriented towards developers: **https://stackoverflow.com/**

END OF FILE